(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification⁷: H04N 7/50

(21) International Application Number: PCT/IB02/00044

(22) International Filing Date: 10 January 2002 (10.01.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
01200088.1   12 January 2001 (12.01.2001) EP
01200206.9   19 January 2001 (19.01.2001) EP

(71) Applicant: KONINKLIJKE PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventor: JASPERS, Egbert, G., T.; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(74) Agent: DUIJVESTIJN, Adrianus, J.; Internationaal Octrooibureau B.V., Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(81) Designated States (national): CN, JP, KR.

(84) Designated States (regional): European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR).

Published:
— with international search report
— before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: UNIT AND METHOD FOR MEMORY ADDRESS TRANSLATION AND IMAGE PROCESSING APPARATUS COMPRISING SUCH A UNIT
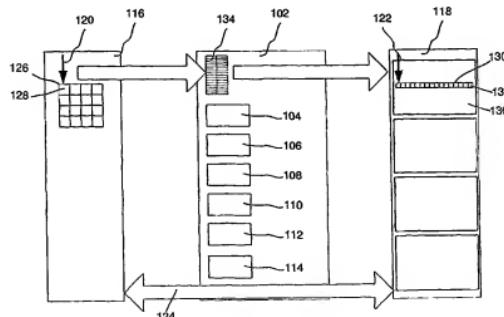
(57) Abstract: A memory device (118) may use a burst access mode to access a number of consecutive data words by giving one read or write command. These data bursts represent non-overlapping data-units in the memory device which can only be accessed as a whole. Because a request for data may contain only a few bytes and can overlay more than one data-unit in the memory device, the amount of transfer overhead is significant. To minimize this overhead a good mapping from logical addresses to physical addresses is important. For the address translation, a logical array is partitioned into a set of rectangles called windows and each window is stored in a row of the memory device. Data request of data-blocks that are actually stored or retrieved, are analyzed during a predetermined period, to calculate the optimal window size. The memory address translation unit (102) performs the analysis and generates the mapping.

WO 02/056600 A1

Unit and method for memory address translation and image processing apparatus comprising such a unit

The invention relates to a memory address translation unit designed to generate a mapping to translate a logical address of a data element of a data-block, to a physical address of a data cell of a data-unit, being part of a memory device, comprising an analyzer for analyzing properties of the memory device and properties of data-blocks, and a mapping generator for generating the mapping based on output of the analyzer.

5   The invention further relates to an image processing apparatus comprising such a memory address translation unit.

The invention further relates to a method to generate a mapping to translate a logical address of a data element of a data-block, to a physical address of a data cell of a data-unit, being part of a memory device, comprising a first step to analyze properties of the

10   memory device and properties of data-blocks, and a second step to generate the mapping based on results of the first step.

15   A unit of the kind described in the opening paragraph is known from the article Array Address Translation for SDRAM-based Video Processing Application, in Visual Communications and Image Processing 2000, Proceedings of SPIE- The International Society for Optical Engineering, Vol. 4067, part two, Year 2000, pages 922-931.

As the resolution of video processing applications becomes high, video signal

20   processors have to deal with a large amount of data within a tightly bounded time. To obtain high memory bandwidth, some memory devices, e.g. SDRAM, use an important feature: the burst access mode. The burst access mode makes it possible to access a number of consecutive data words by giving one read or write command. Because the reading of dynamic memory cells is destructive, the content in a row of cells in the memory bank is

25   copied into a row of static memory cells, the page registers. Subsequently, access to this row is provided. Similarly, when another row has to be accessed, first the content in the row of static memory cells has to be copied back into the original, destructed, dynamic cells. These actions, referred to as row-activations and respectively pre-charges, consume valuable time in which the array of memory cells, i.e. a bank, cannot be accessed. To optimize the utilization

of the memory-bus bandwidth, data should only be accessed at the grain size of a data burst, e.g. eight words. These data bursts represent non-overlapping data-units in the memory device which can only be accessed as a whole. Because a request for data may concern only a few bytes, i.e. the data-units are larger than the requested data blocks and a request for data can involve more than one data-unit in the memory device, the amount of transfer overhead may be significant. To minimize this overhead a good mapping from logical addresses to physical addresses is important. To illustrate this the following example is provided. A video processing algorithm processes two-dimensional arrays of 8x8 pixels. Such two-dimensional arrays are represented as data-blocks. If the addresses of the various pixels are linearly mapped to physical addresses, accessing such a data-block causes seven row-changes. However if the pixels of such 8x8 data-block are kept in one data-unit of the memory device, accessing such a 8x8 data-block does not induce any row-changes.

From the article Array Address Translation for SDRAM-based Video Processing Application, in Visual Communications and Image Processing 2000, Proceedings of SPIE- The International Society for Optical Engineering, Vol. 4067, part two, Year 2000, pages 922-931, is known a memory address translation unit for reducing the number of memory cycles in multi-dimensional video processing applications. In this article an algorithm is described that searches for a suitable window size considering the memory access patterns and memory parameters. A logical array, e.g. a video frame, is partitioned into a set of rectangles called windows. The window size determines how pixels from e.g. a video frame are divided into a number of groups of related pixels. In other words, a video frame is split in a number of regions, wherein the spatial dimensions of such a region correspond to the dimensions of a window. All pixels from such a region belong to one group of related pixels. Each group of related pixels is stored in a row of the memory device. The length of a window corresponds with the number of pixels in horizontal direction. The height of a window corresponds with the number of pixels in vertical direction. Address translation means determination of a physical address for a logical address. To store a data element into a memory device, a physical address of a data-cell, being a part of a data-unit, has to be calculated for the logical address of the data element. Each pixel has a logical address. This address might be the set of co-ordinates of the pixel within the video frame. If it is required that a group of related pixels has to be stored in one data-unit, then this determines the calculation of the physical addresses related to the pixels to be stored. The pixels from a group of related pixels should be mapped to consecutive physical addresses. In the article a mapping of video data into memory is proposed related to analyzing the application software.

The consequence of estimating the window size by analyzing the application software only, is that the estimated window size is not optimal. This results in a mapping of logical to physical addresses that is not optimal. The effect is that a group of related pixels is not stored in one data-unit but spread over several data-units. One data-block request, to access such a group of related pixels has a significant memory transfer overhead. The memory device is invoked several times, in stead of performing one burst access.

Besides the consequence of estimating the window size by analyzing the application software only, without considering data dependencies, it is not always possible to analyze the application software, because the code might not be available. That may be an issue if the code, or parts of it, has been developed by a third party.

It is a first object of the invention to provide a memory address translation unit of the kind described in the opening paragraph with an improved mapping to translate a logical address of a data element of a data-block to a physical address of a data cell of a data-unit.

It is a second object of the invention to provide an image processing apparatus comprising such a memory address translation unit.

It is a third object of the invention to provide a method of the kind described in the opening paragraph with improved mapping to translate a logical address of a data element of a data-block to a physical address of a data cell of a data-unit.

The first object of the invention is achieved in that the analyzer analyzes values of properties of actual data-blocks that are actually stored to or retrieved from the memory device during a predetermined period of time. Values of properties of data-blocks that are actually stored or retrieved runtime, can differ from values of properties of data-blocks from which it is assumed, based on analysis of the application software only, that they will be stored or retrieved. Furthermore the probability of occurrence of the data-blocks is impossible to derive by analyzing the application software, without considering data dependencies. Most application programs contain a number of loops and conditional tests. The consequence of these conditional tests is that the program has a number of parallel paths. The input data of the program determines which paths are actually taken. In other words the input data to be processed by an application program strongly influences the internal variables of the program and thus the memory accesses. This happens for example in an

MPEG decoder. It strongly depends on the strategy taken by the encoder what type of data-blocks the MPEG decoder will have as its operands.

An embodiment of the memory address translation unit according to the invention is described in claim 2. An important property of the data-blocks is the probability distribution of the physical address of each first data cell corresponding to the first data element of each data-block that is actually stored to or retrieved from the memory device. Based on such address, the size of a data-unit and the size of the particular data-block it can be determined how many data-units contain data elements from that data-block. If the number of data elements of one data-block fit in one data-unit it is favorable that they are placed in one data-unit. Spread of data elements over data-units must be as minimal as possible. Because if a request for data overlays more than one data-unit in the memory device, then each of the data-units must be accessed resulting in a significant memory transfer overhead.

An embodiment of the memory address translation unit according to the invention is described in claim 3. Another important property of the data-blocks is the probability of occurrence. A program can have several types of operands corresponding to types of data-blocks. For example in the case of MPEG the set of data-blocks is $V = (16\text{x}16), (17\text{x}16), (16\text{x}17), (17\text{x}17), (16\text{x}8), (18\text{x}8), (16\text{x}9), (18\text{x}9), (17\text{x}8), (17\text{x}9), (16\text{x}4), (18\text{x}4), (16\text{x}5), (18\text{x}5)$. However these types are not all used with the same frequency. The probability of occurrence and thus request for memory access differs per type. For MPEG applications, the reference pictures are written in memory by means of MacroBlocks. Although the amount of write requests is equal, the probability of occurrence is relative to the total amount of request. Hence, the occurrence probability of the write requests highly depends on the amount of data requests for the prediction. The latter, is determined by amongst others, the amount of field and frame predictions, the structure of the Group Of Pictures (GOP), the amount of forward, backward and bi-directional predicted MacroBlocks in a B-picture, etc. It is advantageous if the mapping depends on the probability of occurrence. If the probability of occurrence of a specific type of data-block is relatively high, then it must have a relatively high influence on the window size.

An embodiment of the memory address translation unit according to the invention is described in claim 4. Based on the information gathered by inspecting all memory accesses during a predetermined period of time, the memory translation unit is arranged to analyze whether the current mapping, as used during the predetermined period of time, resulted in the lowest possible memory transfer overhead for the data-blocks stored in

respectively retrieved from the memory device. Besides information about the data-blocks
which is achieved by inspecting all memory accesses during a predetermined period of time,
values of properties of the memory device must be known, e.g. the width of the memory bus
and the number of banks. These properties are constants and must be provided to the memory
5 address translation unit. With a cost function of the memory transfer overhead the optimal
dimensions of the windows can be calculated mathematically. The mapping is primarily
based on the dimensions of the windows.

An embodiment of the memory address translation unit according to the
invention is described in claim 5. The memory address translation unit is arranged to create a
10 new mapping when it is externally triggered to do so. The advantage is that the external
trigger can be invoked at any time.

An embodiment of the memory address translation unit according to the
invention is described in claim 6. The memory address translation unit is arranged to create a
number of mappings. For each of these mappings the memory address translation unit is able
15 to calculate the memory transfer overhead One of these mappings is the one that is actually in
use, i.e. the active mapping. If the memory address translation unit detects that the active
mapping did not result in the lowest possible memory transfer overhead, then the mapping
with the lowest possible memory transfer overhead can be made the active mapping. The
result of this strategy is that at any time the chance that the active mapping is equal to the
20 mapping with the lowest possible memory transfer overhead is relatively high.

An embodiment of the memory address translation unit according to the
invention is described in claim 7. The memory address translation unit is beneficial in all
cases that use is made of a memory device having the feature of burst access mode. The burst
access mode makes it possible to access a number of consecutive data words by giving one
25 read or write command. An example of such memory device is a synchronous dynamic
random access memory (SDRAM) device. Also for accessing more sophisticated memory
devices like double data rate synchronous DRAM (DDR SDRAM) or Direct Rambus DRAM
the memory address translation unit is beneficial.

An embodiment of the memory address translation unit according to the
30 invention is described in claim 8. Most video processing algorithms are based on multi-
dimensional arrays, i.e. data-blocks and nested loops. Usage of the memory address
translation unit can be very beneficial for video or still-image processing algorithms. In that
case an element of a data-block is related to the luminance value of a pixel. The luminance

value of a pixel may represent the value of the combination of color components, Red Green and Blue, or the value of one of the color components.

An embodiment of the memory address translation unit according to the invention is described in claim 9. Besides the feature of the burst access mode, memory devices can have a multiple bank architecture. To hide the memory cycles needed for row-activations and pre-charges, a multiple bank architecture is used, where each bank is accessed alternatively. A bank is accessed independently from the other banks. The organization into memory banks, i.e. a strategy to spread the data-blocks over the various banks, is an important element for memory bandwidth efficiency. This strategy must be provided to the memory address translation unit. Knowledge of the meaning of the data-blocks is important. For several applications in a multi-media system, it is necessary to read the video data both progressively and interlaced, e.g. frame prediction and field prediction for MPEG decoding. However, when subsequent odd and even lines are mapped onto the same data-unit, it is not possible to access only odd or even lines without wasting memory bandwidth. Therefore, the odd and even lines are positioned in different banks of the memory device. As a result, the data-units are interleaved in the memory device when the vertical size of the data-blocks is larger than 1. This knowledge must be taken into account in order to determine the optimal mapping. To achieve this, the set of data-blocks used to determine the optimal mapping must consist of data-blocks belonging to data request of:

- both progressive and interlaced video, or
- interlaced video only, or
- progressive video only

This means that the set of data-blocks comprises:

- data-blocks for which a data element of a data-block is related to the luminance value of a pixel belonging to an even line of a video frame;
- data-blocks for which a data element of a data-block is related to the luminance value of a pixel belonging to an odd line of a video frame; and
- data-blocks for which a data element of a data-block is related to the luminance value of a pixel belonging to an even line or an odd line of a video frame.

An embodiment of the memory address translation unit according to the invention is described in claim 10. In many cases data-blocks will be requested, wherein all rows of the data-block have equal numbers of data elements. In that case all data elements of the rectangular data-block are accessed. It might be that retrieval of all data elements of this rectangular data-block is not required, e.g. because some of these data elements have been

accessed already. The memory address translation unit is able to generate a mapping by analyzing data-blocks of both types.

The third object of the invention is achieved in that the method comprises a first step to analyze values of properties of the data-blocks that are actually stored to or retrieved from the memory device during a predetermined period of time and a second step to generate the mapping.

These and other aspects of the invention will become apparent from and will be elucidated with reference with respect to the implementations and embodiments described hereinafter and with reference to the accompanying drawing, wherein:

Figure 1 schematically shows the main modules of the memory address translation unit and the main components to which the memory address translation unit is connected;

Figure 2A schematically shows the mapping of 64x1 pixels onto memory device data-units;

Figure 2B schematically shows the mapping of 16x4 pixels onto memory device data-units;

Figure 3 schematically shows a mapping of interlaced video onto memory device data-units; and

Figure 4 shows the most important elements of an image processing apparatus according to the invention.

Figure 1 schematically shows the main modules of the memory address translation unit 102 and the main components to which the memory address translation unit is connected. The processor 116 requests for data accesses. The data is stored in the memory device 118. Each request for data access by the processor results in a data transfer 124 from the processor to the memory device or vice versa. The processor provides the logical address 120 of each data element 128 of each data-block 126, to the memory address translation unit, with each request for data access. The memory address translation unit translates this logical addresses 120 to a physical addresses 122. The memory address translation unit provides the physical address to the memory device. The memory device 118 contains a number of data-

units 130. Each data-unit contains a number of data-cells 132. The memory device comprises
4 banks 136.

The memory address translation unit comprises the following components:

- A data-block analyzer 104. The data-block analyzer is designed to determine

5   values of properties of each data-block that is stored or retrieved. The properties of a data-
block are for example the vertical size and the horizontal size, the physical address of the first
data cell corresponding to the first data element of the data-block. The data-block analyzer is
designed for checking the type of video, i.e. interlaced or progressive. Based on collected
values of properties of the data-blocks that have been stored or retrieved during a

10   predetermined period of time the data-block analyzer 104 can derive values of the
probabilities of occurrence of certain data requests; the probability that a data-block with
certain dimensions is accessed or the probability distribution of the physical addresses of
each first data element of each data-block.

- A memory transfer overhead calculator 106. The memory transfer overhead

15   calculator is designed to calculate the memory transfer overhead for a set of control
parameters. The information that is gathered by the data-block analyzer 104, is input for the
memory transfer overhead calculator is 106. Besides that information, properties of the
memory device must be known, e.g. the width of the memory bus and the number of banks
136. These properties are constants. The organization into memory banks, i.e. a strategy to

20   spread the data-blocks over the various banks 136, is an important element for memory
bandwidth efficiency. This strategy must be provided to the memory transfer overhead
calculator.

- A minimum cost establisher 108. The minimum cost establisher provides the
memory transfer overhead calculator 106 with various sets of control parameters. The

25   minimum cost establisher is arranged to determine which set of control parameters results in
the lowest possible memory transfer overhead. Output from the minimum cost establisher
comprises the optimum window size. Another task of the minimum cost establisher is to
verify whether another mapping would have resulted in a lower memory transfer overhead, if
the mapping had been used during a predetermined period of time instead of the current

30   mapping, for the data-blocks stored in respectively retrieved from the memory device. The
minimum cost establisher 108 notifies the mapping generator 110, if the minimum cost
establisher detects that the current mapping is not the most optimal mapping.

- A mapping generator 110. The mapping generator is arranged to generate the
mapping to translate a logical address of a data element of a data-block to a physical address

of a data cell of a data-unit. To generate this mapping the mapping generator 110 requires information that is calculated by the minimum cost establisher 108. The mapping generator is subscribed on events, e.g. better mapping found, generated by the minimum cost establisher. The output from the mapping generator is a look up table 134. This look up table 134

5 describes the mapping.

- An address generator 112. The address generator determines for each instance of a logical address the physical address. It uses the look up table 134.

- A memory command generator 114. To access a data-unit in the memory device, e.g. SDRAM, first a row-activate command also called Row Address Strobe (RAS)

10 has to be issued for a bank 136 to copy the addressed row into the page of that bank. After some delay, a read or write command also called Column Address Strobe (CAS) for the same bank can be issued to access the required data-units in the row. When all required data-units in the row are accessed, the corresponding bank can be pre-charged. The timing of all these commands is critical. The memory command generator, creates these commands for each

15 data access, in the right order and with the right delay in between the commands.

Figure 2A schematically shows the mapping of 64x1 pixels onto memory device data-units and Figure 2B schematically shows the mapping of 16x4 pixels onto memory device data-units. It is assumed that one pixel corresponds with one byte. The memory device 201 comprises 64 data-units. Each data-unit can contain 64 bytes. The logical

20 size of the memory device is such that it can keep the pixels from 32 video lines with 128 pixels each. The memory device contains 4 banks. The data-units corresponding to the various banks are indicated with references 202-208. For the mapping of pixels, several options can be recognized. The most straight forward way is to map 64 successive pixels of a video line onto one data-unit as depicted in Figure 2A. Figure 2A shows how each

25 consecutive row of 64 pixels is interleaved in the banks in both horizontal and vertical direction. Due to the interleaved mapping, the accesses to the memory nicely address the four banks successively if the pixel data is sequentially read or written. However, when a data-block of 16x16 pixels is requested from the memory device, the amount of data that is transferred is much more. If the data-block is horizontally positioned within one data-unit,

30 64x16 pixels are transferred. If the data-block overlays two data-units in horizontal direction, the amount of transferred data is 128x16 pixels. When a mapping strategy is chosen as depicted in Figure 2B, the overhead is less. However, when a data-block of 128x1 is requested, Figure 2A provides a better mapping strategy.

Figure 3 schematically shows a mapping of interlaced video onto memory
data-units. It is assumed that one pixel corresponds with one byte. The memory device 301
comprises 64 data-units. Each data-unit can contain 64 bytes. The logical size of the memory
device is such that it can keep the pixels from 32 video lines with 128 pixels each. The

5   memory device contains 4 banks. The data-units corresponding to the various banks are
indicated with references 302-308. For several applications in a multi-media system, it is
necessary to read the video data both progressively and interlaced, e.g. frame prediction and
field prediction for MPEG decoding. However, when subsequent odd and even lines are
mapped onto the same data-unit, it is not possible to access only odd or even lines without

10 wasting memory bandwidth. Therefore, the odd and even lines are positioned in different
banks of the memory device. As a result, the data-units are interleaved in the memory device
when the vertical size of the data-blocks is larger than 1. The resulting mapping strategy for
data-units of 16x4 pixels is shown in Figure 3. For efficient access of interlaced video data,
the mapping of odd and even video lines onto odd and even banks is toggled after four units

15 in vertical direction. Thus the first odd lines are mapped onto bank 0 and 2 while the even
lines are mapped onto bank 1 and 3. In the successive video lines it is done the opposite way;
the odd lines are mapped onto bank 1 and 3 and the even lines are mapped onto bank 0 and 2.
For progressive video this does not make a difference, but for interlaced video however, this
results in addressing of all banks instead of only the odd or only the even banks. Note that for

20 retrieval of data-blocks of 16x4 pixels with progressive video lines the size has become eight
lines in vertical direction, whereas for access to data-blocks with interlaced video the size is
four lines.

Figure 4 shows the most important elements of an image processing apparatus
according to the invention. The image processing apparatus 402 has a processing means 406

25 for processing data representing images to be compressed, de-compressed, enhanced or
filtered. This data may be broadcasted and received via an antenna or cable but may also be
data from a storage device like a VCR (Video Cassette Recorder) or DVD (Digital Versatile
Disk). The interface unit for importing data 410 has a connector 414. The interface unit for
importing data is connected to a bus 412 for data transfer inside the image processing

30 apparatus 402. The data can be sent out via a cable but may also be stored my means of a
device like a VCR or CD- Recorder (Compact Disk Recorder). The interface unit for
exporting data 418 has a connector 416. The interface unit for exporting data is connected to
the bus 412 for data transfer inside the image processing apparatus 402. The data may also be
generated by the image processing apparatus by means of an image capture unit 420. The

data may also be visualized by the image processing apparatus by means of an image display unit 422. The data can be stored in the memory device 408. Access to data to be stored or retrieved in respectively from the memory device is handled by the memory address translation unit 404. The interface unit for receiving data 410, the interface unit for exporting

5   data 418 and the processing means 406 communicate with the memory address translation unit 404 in order to access data.

In Figure 2A, Figure 2B and Figure 3 examples are provided of mappings. To find an optimal mapping any minimization method can be used to determine M the horizontal size of the window and N the vertical size of the window such that the following

10 condition is satisfied:

$$\min_{M,N \in \aleph^+ \wedge M \cdot N = S} \left(\bar{o}(M,N,V) \right), \tag{1}$$

where S is the size of the window given in amount of pixels, $\aleph^+ = \{1,2,3,4,5,....\}$ and

15 $$\bar{o}(M,N,V) = \bar{o}_i(M,N,V_i) + \bar{o}_p(M,N,V_p) \tag{2}$$

with $V = V_i \cap V_p$.

$\bar{o}_i(M,N,V_i)$ and $\bar{o}_p(M,N,V_p)$ represent the overhead for respectively interlaced data-block requests and progressive data-block requests. When a set of data-blocks, consisting of both interlaced and progressive video, has to be considered, the set has

20 to be separated and Equation 3 respectively Equation 4 have to be applied to the subsets.

The overhead for interlaced data-block requests can be calculated as follows:

$$\bar{o}_i(M,N,V_i) = \left(\frac{\sum_{B_x \times B_y \in V_i} P(B_x \times B_y) \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left(P_{B_x \times B_y}(M,N) \cdot \text{hsize}(B_x) \cdot \text{vsize}_i(B_y) \right)}{\sum_{B_x \times B_y \in V_i} P(B_x \times B_y) \cdot B_x \cdot B_y} \right) \cdot 100\%, \tag{3}$$

with,

$$\text{hsize}(B_x) = 1 + \left\lfloor \frac{B_x + m - 1}{M} \right\rfloor, \text{ and } \text{vsize}_i(B_y) = 1 + \left\lfloor \frac{B_y + n - 1}{N} \right\rfloor$$

25 Where V_i is the set of possible data-blocks $B_x \times B_y$ and $P(B_x \times B_y)$ the probability of the data-block. And where $P_{B_x \times B_y}(M,N)$ is equal to the probability that the upper left corner pixel, i.e. the first element of a data-block, of a requested data-block $B_x \times B_y$ is positioned at any location (x,y) that satisfies the following condition: $x \bmod M = m$ AND $y \bmod N = n$.

The numerator in equation 3 represents the amount of transferred data including overhead. The denominator represents the amount of requested data without the overhead. The overhead for progressive data-block requests can be calculated as follows:

$$\overline{o}_p(M,N,V_p) = \left(\frac{\displaystyle\sum_{B_x \times B_y \in V_p} P(B_x \times B_y) \sum_{m=0}^{M-1}\sum_{n=0}^{2N-1} \left(P_{B_x \times B_y}(M,N) \cdot \text{hsize}(B_x) \cdot \text{vsize}_p(B_y)\right)}{\displaystyle\sum_{B_x \times B_y \in V_p} P(B_x \times B_y) \cdot B_x \cdot B_y} \right) \cdot 100\%, \quad (4)$$

with,

$$\text{hsize}(B_x) = 1 + \left\lfloor \frac{B_x + m - 1}{M} \right\rfloor \text{ and } \text{vsize}_p(B_y) = \left(1 + \left\lfloor \frac{\left\lfloor \frac{B_y}{2} \right\rfloor + \left\lfloor \frac{n}{2} \right\rfloor - 1}{N} \right\rfloor\right) + \left(1 + \left\lfloor \frac{\left\lfloor \frac{B_y}{2} \right\rfloor + \left\lceil \frac{n}{2} \right\rceil - 1}{N} \right\rfloor\right)$$

The numerator in equation 4 represents the amount of transferred data including overhead. The denominator represents the amount of requested data without the overhead. For this optimization method, the data-blocks with $B_y = 1$ are considered as interlaced requests and therefore, they are contained in set V_I.

It should be noted that the above-mentioned embodiments illustrate rather than limit the invention and that those skilled in the art will be able to design alternative embodiments without departing from the scope of the appended claims. In the claims, any reference signs placed between parentheses shall not be constructed as limiting the claim. The word 'comprising' does not exclude the presence of elements or steps other than those listed in a claim. The word "a" or "an" preceding an element does not exclude the presence of a plurality of such elements. The invention can be implemented by means of hardware comprising several distinct elements and by means of a suitable programmed computer. In the unit claims enumerating several means, several of these means can be embodied by one and the same item of hardware.

CLAIMS:

1.   A memory address translation unit (102) designed to generate a mapping (134) to translate a logical address (120) of a data element (128) of a data-block (126), to a physical address (122) of a data cell (132) of a data-unit (130), being part of a memory device (118), comprising an analyzer for analyzing properties of the memory device and properties of data-blocks, and a mapping generator for generating the mapping based on output of the analyzer, characterized in that the analyzer analyzes values of properties of actual data-blocks that are actually stored to or retrieved from the memory device during a predetermined period of time.

2.   A memory address translation unit as claimed in Claim 1, characterized in that the memory address translation unit is arranged to generate the mapping by analyzing a probability distribution of the physical address of each first data cell corresponding to the first data element of each data-block that is actually stored to or retrieved from the memory device.

3.   A memory address translation unit as claimed in Claim 1, characterized in that the memory address translation unit is arranged to generate the mapping by analyzing probabilities of occurrence of data-blocks.

4.   A memory address translation unit as claimed in Claim 1, characterized in that the memory address translation unit is arranged to analyze whether the current mapping, as used during a predetermined period of time, resulted in a lowest possible memory transfer overhead for the data-blocks stored in respectively retrieved from the memory device.

5.   A memory address translation unit as claimed in Claim 4, characterized in that the memory address translation unit is arranged to generate a new mapping when the memory address translation unit is notified.

6.   A memory address translation unit as claimed in Claim 4, characterized in that the memory address translation unit is arranged to measure the memory transfer overhead for a test-mapping and is designed to replace the current mapping, when it detects that the test-mapping would have resulted in a lower memory transfer overhead, if that test-mapping had been active during the predetermined period of time instead of the current mapping for the data-blocks stored in respectively retrieved from the memory device.

7.   A memory address translation unit as claimed in Claim 1, characterized in that the memory device is a synchronous dynamic random access memory device.

8.   A memory address translation unit as claimed in Claim 1, characterized in that a data element of a data-block is related to the luminance value of a pixel.

9.   A memory address translation unit as claimed in Claim 8, characterized in that the memory address translation unit is arranged to generate the mapping based on data-blocks, used for the analyses of values of properties of actual data-blocks, that can be categorized in:
          - data-blocks for which a data element of a data-block is related to the luminance value of a pixel belonging to an even line of a video frame;
          - data-blocks for which a data element of a data-block is related to the luminance value of a pixel belonging to an odd line of a video frame; and
          - data-blocks for which a data element of a data-block is related to the luminance value of a pixel belonging to an even line or an odd line of a video frame.

10.   A memory address translation unit as claimed in Claim 1, characterized in that the memory address translation unit is arranged to generate the mapping based on data-blocks, used for the analyses of values of properties of actual data-blocks, that can be categorized in:
          - structures of rows of data-elements with a variable number of data elements per row;
          - structures of rows of data-elements with equal number of data elements per row.

11.   An image processing apparatus (402) comprising:

- a memory address translation unit (404) designed to generate a mapping to translate a logical address of a data element of a data-block, to a physical address of a data cell of a data-unit, being part of a memory device, comprising an analyzer for analyzing properties of the memory device and properties of data-blocks, and a mapping generator for generating the mapping based on output of the analyzer, characterized in that the analyzer analyzes values of properties of actual data-blocks that are actually stored to or retrieved from the memory device during a predetermined period of time,

- a processing means (406), and

- a memory device (408).

12.   A method to generate a mapping to translate a logical address of a data element of a data-block, to a physical address of a data cell of a data-unit, being part of a memory device, comprising a first step to analyze properties of the memory device and properties of data-blocks, and a second step to generate the mapping based on results of the first step, characterized in that in the first step values of properties of actual data-blocks, that are actually stored to or retrieved from the memory device during a predetermined period of time, are analyzed.
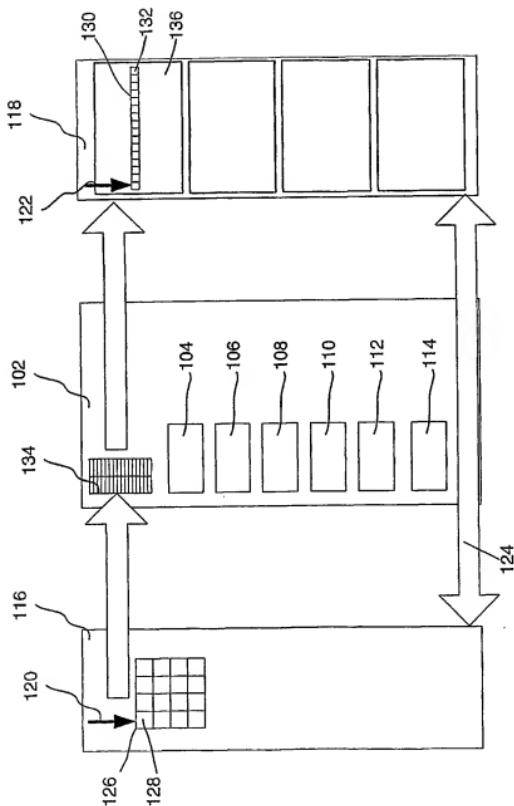
FIG. 1
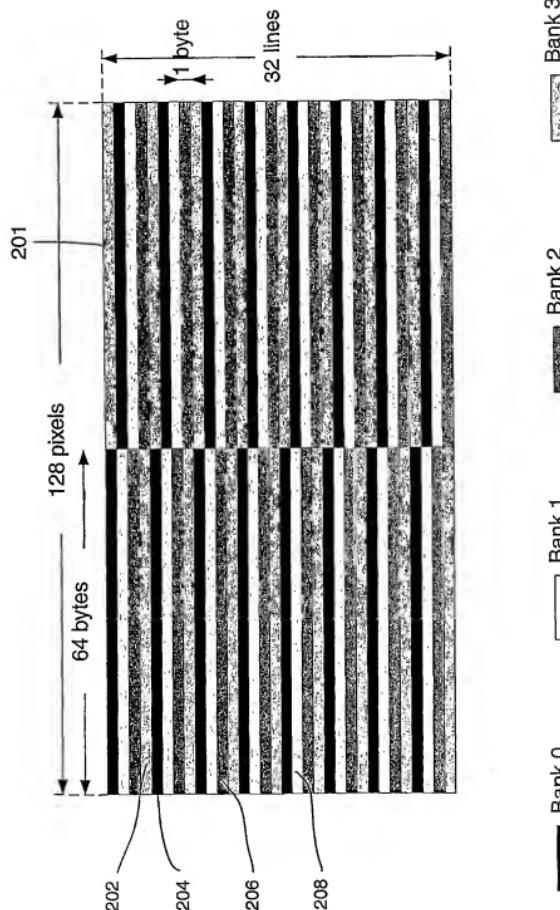
FIG. 2A
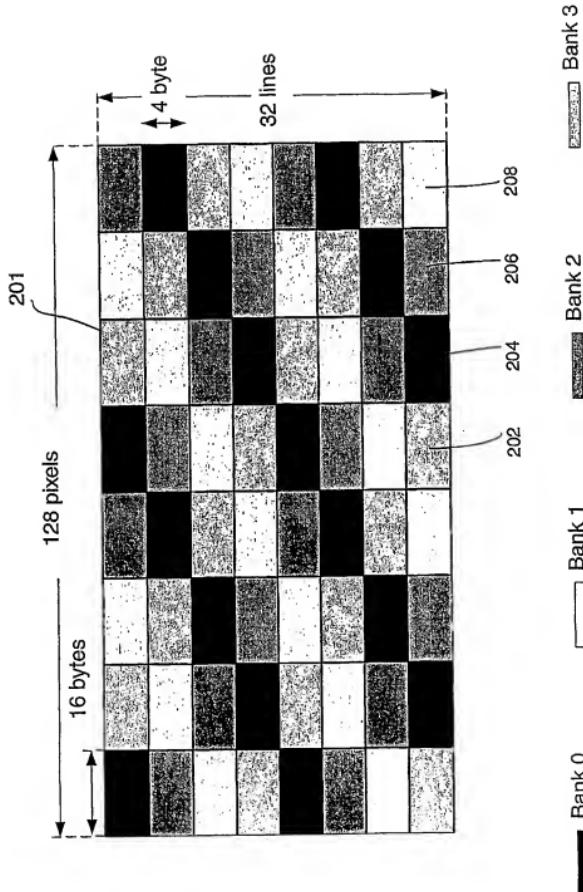
FIG. 2B
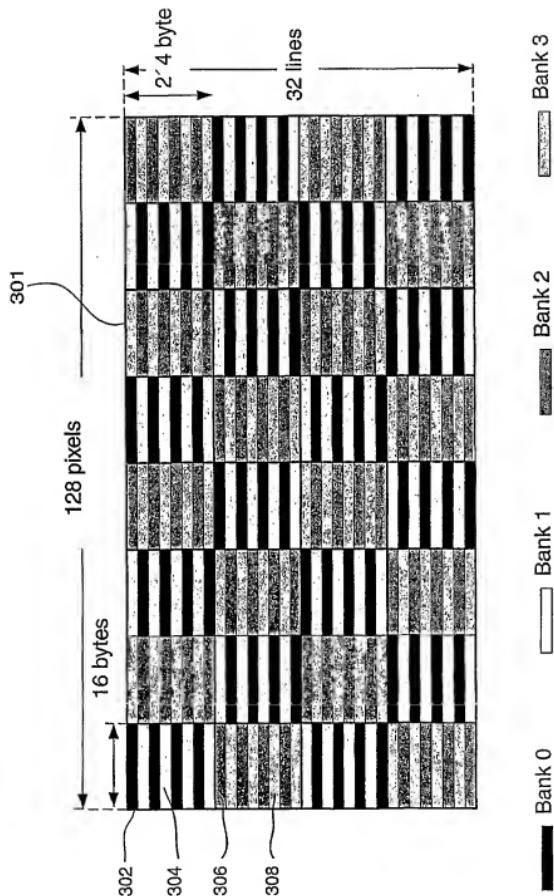
FIG. 3
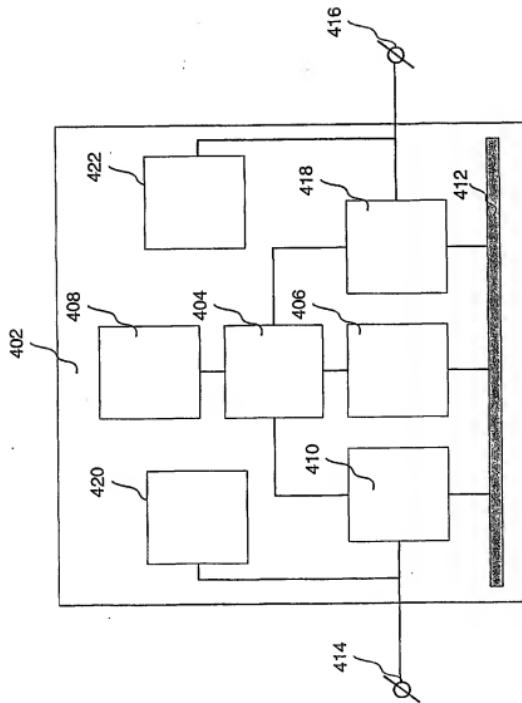
FIG. 4

INTERNATIONAL SEARCH REPORT

Intern...l Application No

PCT/IB 02/00044

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7   H04N7/50

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7   H04N   G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	HANSOO K. ET AL: "Array address translation for SDRAM-based video processing applications" VISUAL COMMUNICATIONS AND IMAGE PROCESSING 2000 - PROCEEDINGS OF SPIE VOL 4067, PERTH, WA, AUSTRALIA, 20 - 23 June 2000, pages 922-931, XP008003690 cited in the application the whole document —/—	1-12

[X] Further documents are listed in the continuation of box C.   [ ] Patent family members are listed in annex.

Date of the actual completion of the international search	Date of mailing of the international search report
24 May 2002	05/06/2002

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Nielsen, O

Form PCT/ISA/210 (second sheet) (July 1992)

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	WINZKER M ET AL: "ARCHITECTURE AND MEMORY REQUIREMENTS FOR STAND-ALONE AND HIERARCHICAL MPEG2 HDTV-DECODERS WITH SYNCHRONOUS DRAMS" 1995 IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS). SEATTLE, APR. 30 - MAY 3, 1995, NEW YORK, IEEE, US, vol. 1, 30 April 1995 (1995-04-30), pages 609-612, XP000583293 ISBN: 0-7803-2571-0 page 611, left-hand column, line 1 -page 612, left-hand column, line 7	1,7-9, 11,12
A	KIM J-M ET AL: "COST-EFFECTIVE ARCHITECTURE FOR HDTV VIDEO DECODER IN ATSC RECEIVERS" IEEE TRANSACTIONS ON CONSUMER ELECTRONICS, IEEE INC. NEW YORK, US, vol. 44, no. 4, November 1998 (1998-11), pages 1353-1359, XP000871435 ISSN: 0098-3063 page 1355, right-hand column, paragraph 1 -page 1357, left-hand column, paragraph 3	1,11,12
A	NEWMAN G: "MEMORY MANAGEMENT SUPPORT FOR TILED ARRAY ORGANIZATION" COMPUTER ARCHITECTURE NEWS, ASSOCIATION FOR COMPUTING MACHINERY, NEW YORK, US, vol. 20, no. 4, 1 September 1992 (1992-09-01), pages 22-30, XP000298581 ISSN: 0163-5964 page 25, left-hand column, paragraph 6 -page 27, left-hand column, paragraph 1	1,11,12
A	OHTANI A ET AL: "A MOTION ESTIMATION PROCESSOR FOR MPEG2 VIDEO REAL TIME ENCODING ATWIDE SEARCH RANGE" PROCEEDINGS OF THE CUSTOM INTEGRATED CIRCUITS CONFERENCE. SANTA CLARA, MAY 1 - 4, 1995, NEW YORK, IEEE, US, vol. CONF. 17, 1 May 1995 (1995-05-01), pages 405-408, XP000536833 ISBN: 0-7803-2585-0 page 407, right-hand column, paragraph 5 -page 408, left-hand column, paragraph 3	1,7-12